Virtual Motor with Synthesized Encoder

Sometimes a user will desire to create a virtual motor on Turbo PMAC with a pulse-and-direction output, a synthesized encoder value, or both. This is easy to do with a PMAC2-style Servo IC in a Turbo PMAC system, as is found on any UMAC axis board, in QMAC, or on any board-level Turbo PMAC2.

There are several reasons that a user may want to do this. In one case, the high-speed position-capture and compare circuits for PMAC2 encoder counters are not available with other types of feedback. A user with, for example, high-resolution parallel-format feedback from an interferometer on a real motor can create a virtual motor/encoder that uses Turbo PMAC's position-following feature to track the position of the real motor, create pulse-and-direction signals that feed the Servo IC channel's hardware encoder counter, and use the values of the counter to generate highly accurate position-compare outputs.

Alternately, we may want to generate position-compare pulses based on a virtual vector path. In this case, the position of this virtual motor can be derived from the vector path of the real axes (as in the example Virtual Vector Master Position.

In this example, we have a UMAC system with 19-bit parallel feedback for Motor 1 coming in on Port A of an Acc-14E board with base address $078C00. We want to generate position-compare outputs based on the actual position of Motor 1, but cannot do so directly because parallel feedback does not support this feature.

We create a virtual Motor 4 using the fourth channel of an Acc-24E2A analog axis board, setting this channel up for pulse-and-direction output with this output wrapped back internally to the channel's encoder counter, as if we were driving an open-loop stepper motor. The counter is used both for simulated feedback and to create position-compare outputs. This virtual motor is slaved to the actual position of Motor 1 using Turbo PMAC's automatic position-following feature.

Note that it will often be possible to use the pulse-and-direction output and encoder counter of the same channel as used for the amplifier output (if analog) and the flags (limits, etc.), because these features of the channel are not used for the actual servo function. If this approach were taken in this example, we would use Servo IC 2 Channel 1 instead of Servo IC 2 Channel 4, as we do below.

This example assumes that all clock frequencies (servo, phase, PFM) are at their factory default values. Explanations for the virtual motor setup can be found in the Pulse and Direction section of the Basic Motor Setup chapter of the Turbo PMAC User Manual.

Encoder Conversion Table Setup

```
I8000=$F78C00                    ; Read Acc-14E Port A
I8001=$313000                    ; Filtered (3) 19 bits (13) from bit 0
I8002=256                        ; Max legit change 256 LSBs per cycle
;...                            ; I8003-8 for Motor 2 & 3 feedback
I8009=$C78218                    ; Servo IC 2 Enc 4, no extension
```

Setup for IC2 Chan 4 for PFM Output Wrapped into Counter

```
I7203=2258                       ; Default clock freq's for IC 2
I7246=3                          ; IC2 Chan 4 PFM and analog outputs
I7240=8                          ; Tie IC2 Ch4 PFM to encoder counter
```

Setup for Virtual Motor 4 using IC 2 Chan 4

```
I400=1                           ; Activate Motor 4 calculations
I401=0                           ; No commutation
I402=$7821C                      ; IC2 Chan 4 Pulse Freq output
I403=@I8009                      ; Use IC 2 Enc 4 counter for pos feedback
I404=@I8009                      ; Use IC 2 Enc 4 counter for vel feedback
I405=@I8002                      ; Use ACC-14E Port A pos as master
I406=1                           ; Enable position following
I407=96                          ; I407/I408 is gear ratio
I408=96                          ; 1:1 gear ratio
```

```
I430=700                        ; Simulated loop Kp for 25Hz nat. freq.
I431=0                          ; No derivative gain needed
I432=15050                      ; Vel FF gain for no tracking error
I433=10000                      ; Int gain to assure no steady-state error
I434=0                          ; Int gain on while moving
I435=0                          ; No accel FF gain
I469=16384                      ; Set max freq output
```

The virtual Motor 4 can always be active and following Motor 1's position feedback. We can then use the encoder counter that provides Motor 4's simulated feedback to generate position-compare outputs. In this next section, we set up the virtual motor's compare output for a 5-count width every 20 counts.

Substitutions and Definitions

```
#define SIC2Ch4EncPos  M401      ; Raw encoder counter position
SIC2Ch4EncPos->Y:$078219,0,24,S
#define SIC2Ch4CompPosA  m408    ; Compare A reg
SIC2Ch4CompPosA->Y:$07821F,0,24,S
#define SIC2Ch4CompPosB  m409    ; Compare B reg
SIC2Ch4CompPosB->X:$07821E,0,24,S
#define SIC2Ch4CompIncr  M410    ; Compare auto-increment reg
SIC2Ch4CompIncr->Y:$07821F,0,24,S
#define SIC2Ch4CompInitSta  M411 ; Compare initial state
SIC2Ch4CompInitSta->Y:$07821D,11
#define SIC2Ch4CompInitEna  M412 ; Compare initialize enable
SIC2Ch4CompInitSta->Y:$07821D,12
```

Program Code Section to Set up Compare Circuitry

(This can be in a motion or PLC program)

```
SIC2Ch4CompPosA=SIC2Ch4EncPos+10 ; +10 counts from present
SIC2Ch4CompPosB=SIC2Ch4EncPos-5  ; -5 counts from present
SIC2Ch4CompIncr=20               ; 20-count auto-increment
SIC2Ch4CompInitSta=0             ; Initialize to 0
SIC2Ch4CompInitEna=1             ; Enable initialization
```